



Accessibility

Werner Hänggi, Thomas Jaggi

Inhalt

- Wie verhindert man grundlegende Probleme für eingeschränkte User?
- Wie testet man Barrierefreiheit?
- Wo findet man relevante Informationen?

Agenda

1. **Weshalb Barrierefreiheit?**
2. Zugang mit **Screenreadern**
3. **WCAG 2.1** als Basis
4. **Tastaturbedienbarkeit**
5. **Visuelle Aspekte**
6. Vertiefung **ARIA**
7. **Automatisches Testing** (Auszug)
8. **Demo** wichtiger Webseiten



Über

- **Werner Hänggi**
 - Software-Entwickler
 - Informatiker ETH
- **AdNovum – Swiss Engineering Quality**
 - Kernkompetenz: High-End Security & Software Engineering
 - Gegründet 1988 – Aktiengesellschaft
 - Zürich (Hauptsitz), Bern, Lausanne, Budapest, Ho-Chi-Minh-Stadt, Singapur, Lissabon
 - 600 Mitarbeiter, 70% Ingenieure (ETH/Uni/FH)
 - Stellenangebote:
http://www.adnovum.ch/karriere_entrypoint/stellenangebote.html



Über (2)

- **Thomas Jaggi**
 - Selbständiger Entwickler, <https://responsive.ch>
 - langjähriger Frontend-Lead bei Unic
 - viele Barrierefreiheit-zertifizierte Projekte
 - Fokus auf Entwicklungsprozess
 - Mitgründer des "Accessibility Developer Guide"

1. Weshalb Barrierefreiheit?

«The power of the Web is in its universality. Access by everyone regardless of disability is an essential aspect.»

Tim Berners-Lee, W3C-Direktor und "Erfinder" des Internets

Deshalb

1/5

Bundesamt für Statistik: «Gemäss verschiedenen Quellen kann die Anzahl Menschen mit Behinderungen auf rund 1,8 Millionen geschätzt werden.»

Benutzer mit speziellen Bedürfnissen

- Einschränkungen bei:
 - Sehen
 - Hören
 - Sprache
 - Motorik
 - Kognitive Fähigkeiten
 - ...

Gesetzgebung

- Grundlage:
 - **Behindertengleichstellungsgesetz:** Dienstleistungen des Gemeinwesens müssen auch für Menschen mit Behinderungen zugänglich sein
- Bund:
 - **Behindertengleichstellungsverordnung:** Internetangebote des Bundes müssen so gestaltet sein, dass Menschen mit Behinderungen diese barrierefrei nutzen können
- Kantone und Gemeinden:
 - «In praktisch allen Kantonen fehlen konkrete Richtlinien und Standards für die Umsetzung des BehiG im Bereich Accessibility. Durch das Fehlen kantonaler Richtlinien herrscht in den Gemeinden eine grosse Unsicherheit.»

Meanwhile in the US

Beyoncé's Parkwood Entertainment sued over website accessibility

A lawsuit claims beyonce.com violates the Americans With Disabilities Act by failing to accommodate visually impaired users



Supreme Court lets blind man sue Domino's over website accessibility

By ignoring Domino's appeal, the Supreme Court opens up the door for lawsuits challenging websites and mobile apps that aren't accessible to people with disabilities.

Reminder: Airlines Should Be Actively Working on Digital Accessibility

Written by: **Bill Curtis-Davidson**

Website accessibility requirements are officially in effect for airlines operating flights within or to the U.S. or selling tickets to the U.S. public. Under the [Air Carrier Access Act \(ACAA\)](#), these airlines are required to ensure that the public-facing content of their websites conforms to the [Web Content Accessibility Guidelines \(WCAG\) 2.0](#) Level AA.

Zertifizierung

Stiftung «Zugang für alle»:
unabhängige Zertifizierungsstelle für barrierefreie Websites

<https://www.access-for-all.ch>



**Zugang für alle
Accès pour tous
Accesso per tutti
Access for all**

Wie barrierefrei ist das Web?

- Auszug aus Auswertung des HTTP Archive für 2019 (**5.8 Mio** Webseiten):
 - 4 out of every 5 sites have text which easily blends into the background, making it unreadable.
 - Almost one out of every three sites on mobile disable [zooming].
 - 49.91% of pages still fail to provide alt attributes for some of their images, and 8.68% of pages never use them at all.
 - Only 24.5% of pages with tables were found to markup their tables with either of these [accessible] methods.
 - Only about a quarter (24.39%) of pages that use buttons or links include textual labels with these controls.
 - Only 22.33% of pages provide labels for all their form inputs.



2. Zugang mit Screenreadern

Beispiel mit **VoiceOver**

VoiceOver for OS X.
A feature that speaks
for itself.

Demo 1: Tabellen

- Take home:
 - **Verwende <th> für Titelzellen (mit scope Attribut, wenn sinnvoll)**
 - **Verwende <caption> für Beschreibung der Tabelle**
- Links:
 - http://cas-fe.github.io/17-Accessibility/1-1_table-inaccessible.html
 - http://cas-fe.github.io/17-Accessibility/1-2_table-accessible.html

Demo 2: Icon-Link

- Take home:
 - **Für rein visuelle Elemente immer einen versteckten alternativen Text definieren**
- Links:
 - https://cas-fe.github.io/17-Accessibility/5-3_link-inaccessible.html
 - https://cas-fe.github.io/17-Accessibility/5-4_link-accessible.html

3. WCAG 2.1 als Basis

- WCAG: **Web Content Accessibility Guidelines**
- W3C Recommendation, <http://www.w3.org/TR/WCAG/>

«Following these guidelines will make content **accessible to a wider range of people with disabilities**, including blindness and low vision, deafness and hearing loss, learning disabilities, cognitive limitations, limited movement, speech disabilities, photosensitivity and combinations of these. Following these guidelines will also often make your Web content **more usable to users in general.**»

Wie die WCAG strukturiert sind

- **4 Prinzipien**
 - *perceivable / wahrnehmbar*
 - *operable / bedienbar*
 - *understandable / verständlich*
 - *robust*
- **13 Richtlinien**
- Testbare **Erfolgskriterien** für jede Richtlinie
- **3 Konformitätslevels** für Erfolgskriterien: A (tiefstes), AA, AAA (höchstes)
- **Ausreichende und empfohlene Techniken** zur Erfüllung eines Kriteriums

WCAG-Beispiel: Labels in Formularen

- Beispiel eines Erfolgskriteriums im Zusammenhang mit Formularen:
 - **Principle 3: Understandable**: Information and the operation of user interface must be understandable.
 - **Guideline 3.3 Input Assistance**: Help users avoid and correct mistakes.
 - **3.3.2 Labels or Instructions**: Labels or instructions are provided when content requires user input. (Level A)
 - **Sufficient Techniques for 3.3.2 - Labels or Instructions**
 - **G131: Providing descriptive labels**
 - **H90: Indicating required form controls using label or legend**

Demo 3: Formular-Markup und -Validierung

- Take home:
 - **Verwende ein <label> für jedes Feld**
 - **Markiere obligatorische Felder** mit Text innerhalb des Labels.
Verwende required and aria-required Attribute als *progressive enhancement*.
 - **Teile dem User mit, wenn ein Input nicht valide ist:**
 - a) Ergänze das Label mit einer Fehlermeldung und fokussiere das erste invalide Feld.
Verwende aria-invalid zur zusätzlichen Indikation.
 - b) Verwende aria-describedby um Fehlermeldungen mit dem entsprechenden Feld zu verbinden.
- Links:
 - http://cas-fe.github.io/17-Accessibility/2-1_form-inaccessible.html
 - http://cas-fe.github.io/17-Accessibility/2-2_form-accessible.html
 - http://cas-fe.github.io/17-Accessibility/2-3_form-aria.html
 - http://cas-fe.github.io/17-Accessibility/2-4_form-html5.html

Demo 4: Seitenstruktur

- Take home:
 - **Verwende versteckte <hX> Elemente, um Bereiche auszuzeichnen**
 - **Verwende die semantisch korrekten Markup-Elemente in Kombination mit ARIA-Rollen. Bsp: <main role="main">...</main>**
 - **Verwende Skiplinks, um dem User zu ermöglichen, Bereiche wie die Navigation zu überspringen**
- Links:
 - http://cas-fe.github.io/17-Accessibility/3-1_structure-inaccessible.html
 - http://cas-fe.github.io/17-Accessibility/3-2_structure-accessible.html



4. Tastaturbedienbarkeit: Demos 5 / 6

- Take home:
 - **Entferne die Focus-Styles nie**
 - Falls doch: Definiere alternative, gut sichtbare Styles
 - **Verwende bereits barrierefreie Elemente wie <a> oder <button>**
 - Falls nicht: Füge role="button" und tabindex="0" hinzu und definiere **keydown listeners** für Space- and Enter-Taste
- Links:
 - http://cas-fe.github.io/17-Accessibility/4-1_focus-inaccessible.html
 - http://cas-fe.github.io/17-Accessibility/4-2_focus-accessible.html
 - http://cas-fe.github.io/17-Accessibility/5-1_button-inaccessible.html
 - http://cas-fe.github.io/17-Accessibility/5-2_button-accessible.html

5. Visuelle Aspekte

- Wichtige WCAG-Richtlinie: «1.4 Distinguishable: Make it easier for users to see and hear content including separating foreground from background.»
- Kriterien (Auswahl):
 - 1.4.1 Use of Color: **Color is not used as the only visual means** of conveying information, indicating an action, prompting a response, or distinguishing a visual element. (Level A):
 - G183: Using a contrast ratio of 3:1 with surrounding text and providing additional visual cues on focus for links or controls where color alone is used to identify them
 - 1.4.3 Contrast (Minimum): The visual presentation of text and images of text has a **contrast ratio of at least 4.5:1**, except for the following: (Level AA) ...

Beispiel: Finde den Link

- «Achromatopsia»-Demo mit «NoCoffee Vision Simulator» in Chrome

The screenshot shows a web browser window with a news article. The title of the article is "Neuer CAS „Front-End-Engineering“". The text discusses the initiation of a new certificate program in collaboration with leading web service providers. It mentions that from May 2014, the HSR will offer a postgraduate course in "Front-End-Engineering" for software developers and users. The article also notes the introduction of a six-month certification course.

News 18.03.2014 08:53

Neuer CAS „Front-End-Engineering“

Die HSR initiiert zusammen mit führenden Webdienstleistern einen topaktuellen Zertifikatslehrgang „Front-End-Engineering“. Ab Mai 2014 bietet die HSR den erstmaligen Weiterbildungskurs [CAS Front-End-Engineering](#) an. Das neuste Lehrangebot richtet sich an Software Entwicklerinnen und Entwickler, die moderne Usability-Methoden erlernen und die Benutzung der heterogenen Endgeräte wie Smartphones, Tablets oder Desktops zu einem positiven Frontend-Erlebnis für die User machen.

Mit der Einführung des sechsmonatigen Zertifizierungslehrgangs Certificate of

The screenshot shows the same news article from the previous screenshot, but it is displayed in grayscale. This represents the "NoCoffee Vision Simulator" effect, which simulates color blindness or achromatopsia. The HSR logo and navigation menu are visible on the left, and the main content area shows the news article in grayscale.

HSR Hochschule für Technik 18.03.2014 08:53

Neuer CAS „Front-End-Engineering“

Die HSR initiiert zusammen mit führenden Webdienstleistern einen topaktuellen Zertifikatslehrgang „Front-End-Engineering“. Ab Mai 2014 bietet die HSR den erstmaligen Weiterbildungskurs [CAS Front-End-Engineering](#) an. Das neuste Lehrangebot richtet sich an Software Entwicklerinnen und Entwickler, die moderne Usability-Methoden erlernen und die Benutzung der heterogenen Endgeräte wie Smartphones, Tablets oder Desktops zu einem positiven Frontend-Erlebnis für die User machen.

Mit der Einführung des sechsmonatigen Zertifizierungslehrgangs Certificate of

Kontrast messen

- Online-Tool: <https://webaim.org/resources/contrastchecker/>
- Applikation: <https://developer.paciellogroup.com/resources/contrastanalyser/>
- Gute Übersicht zum Thema: <https://www.smashingmagazine.com/2014/10/color-contrast-tips-and-tools-for-accessibility/>

Skalierbarkeit

- Ansatz: relative Dimensionen wie em oder %
 - **WCAG: 1.4.4 Resize text:** Except for captions and images of text, text can be resized without assistive technology up to 200 percent without loss of content or functionality. (Level AA)
- Anti-Pattern:

```
<meta name="viewport" content="width=device-width,  
initial-scale=1.0, user-scalable=no" />  
<meta name="viewport" content="width=device-width,  
initial-scale=1.0, maximum-scale=1" />
```
- Okay:

```
<meta name="viewport" content="width=device-width,  
initial-scale=1.0" />
```

Der Browser ist
tendenziell intelligenter als wir.

6. Vertiefung (WAI)-ARIA

- Web Accessibility Initiative:
Accessible Rich Internet Applications
- W3C Recommendation:
<https://www.w3.org/TR/wai-aria/>
- *Ziel:* Inhalte (insbesondere dynamische) zugänglich machen, welche standardmäßig nicht zugänglich sind

ARIA - when HTML simply is not enough

There are indeed situations where standard HTML does not provide functionalities for all the requirements developers may have for implementing modern and interactive websites. For this, the Accessible Rich Internet Applications (ARIA) technical specification was introduced. In this chapter, we introduce how it is purposely used, show its potential and shortcomings as well as alternatives to it.

ARIA: Grundlagen

- spezielle **Attribute** im Markup
- **kein** Einfluss auf Verhalten des Browsers, nur auf die Informationen, welche er an den Screenreader weiterleitet

ARIA: Roles

- semantische Definition für ein Element
- «Attaching a role gives assistive technologies information about how to handle each element.»
- gewisse Elemente haben **implizite** Rollen:
 - Beispiel: <nav> hat implizit die ARIA-Rolle "navigation"
 - Sollte je nach Browser-Support trotzdem explizit hinzugefügt werden
(für ältere Browser, welche <nav> nicht nativ unterstützen)

ARIA: States and Properties

- Schwierige Abgrenzung, teilweise unter gemeinsamem Begriff "Attribut"
 - Properties ändern tendenziell seltener während «life-cycle» der Applikation
-
- State: aria-expanded, aria-hidden
 - Property: aria-describedby, aria-live

Demo 7: Auto-suggest

- Take home:
 - **Definiere keydown listeners für Pfeiltasten, um durch Suggestions zu navigieren**
 - **Informiere den User mit ARIA live region über dynamischen Inhalt**
 - **Verwende die Rollen listbox (Container) und option (Suggestions)**
 - **Verwende den Zustand aria-expanded je nach Sichtbarkeit der Suggestions**
- Links:
 - http://cas-fe.github.io/17-Accessibility/6-1_autosuggest-inaccessible.html
 - http://cas-fe.github.io/17-Accessibility/6-2_autosuggest-accessible.html

Demo 8: Modal

- Take home:
 - **Stelle sicher, dass der Fokus innerhalb des Modals bleibt**
 - **Verwende role=“dialog” für den Container und verknüpfe ihn via aria-labelledby mit dem Titel**
 - **Schliesse den Dialog auf esc**
- Links:
 - http://cas-fe.github.io/17-Accessibility/6-3_modal-inaccessible.html
 - http://cas-fe.github.io/17-Accessibility/6-4_modal-accessible.html

JavaScript-Frameworks und ARIA

- React, Vue, Angular sind *grundsätzlich barrierefrei*
- Relevant ist der UI-Code
- Verbreitete UI-Frameworks sind oft nicht wirklich barrierefrei
- Positivbeispiele:
 - <https://reach.tech>
 - <https://bootstrap-vue.org>
 - <https://material.angular.io> (Spielraum nach oben)

Ally.js: Nützliche Accessibility-Helpers

- “Focus trapping” ist in vielen Situation wichtig:
 - <https://github.com/edenspiekermann/a11y-dialog> (vorherige Demo) kümmert sich selber darum
 - **Ally.js** hilft bei custom Modulen: <https://allyjs.io/api/maintain/tab-focus.html>
- Weitere hilfreiche Features:
 - Unterscheidung zwischen Maus und Tastatur als “**Focus source**”: <https://allyjs.io/api/style/focus-source.html>
 - Detektierung, ob ein Element sichtbar ist: <https://allyjs.io/api/is/visible.html>

ally.js

JavaScript library to help modern web applications with accessibility concerns by making accessibility simpler

Accessibility Developer Guide: Code-Beispiele



- <https://www.accessibility-developer-guide.com>
- <https://github.com/Access4all/adg/>

“The «Accessibility Developer Guide» is an initiative of «Access for all», Swiss Foundation for technology adapted to people with disabilities.

It is developed and maintained in collaboration with a number of acclaimed web agencies”

Slack: https://join.slack.com/t/a11y-dev-guide/shared_invite/enQtMzMwOTkxNTI3NDYwLWI3MjRmZDQwOGMxOGVhNTI2NDq0ODhiMDUyZTQ4MDE4MzU1NmZiMTY2YmNjNjliYWEzODhjZDYwYjA1MDU4NmU

7. Automatisches Testing: Auszug

- HTML-Linting nach Accessibility-Kriterien:
 - OSS: Beispiel [HTML_CodeSniffer](#)
 - Gulp-Wrapper: <https://github.com/yargalot/gulp-accessibility>
 - Demo:
 - <https://github.com/cas-fe/gulp-accessibility> klonen und "npm run demo" ausführen
 - Optionen für "accessSniff-json" in "gulpfile.js" anpassen



- SAAS: Beispiel: <https://tenon.io>
- Browser Extensions:
 - aXe: <https://www.deque.com/axe/>
 - Wave: <https://wave.webaim.org/extension/>



Automatisches Testing: Trefferquote

- Beispiel mit Tabelle:
 - Nur **Tenon** warnt explizit:

```
<table>
<tbody><tr>
<td>
<strong>Time</strong>
</td>
<td>
<strong>Design Track</strong>
</td>
<td>
```

Test ID 34

Warning / priority 100%
This table does not have any headers.

(line: 17, 5)

Web Content Accessibility Guidelines (WCAG) 2.0, Level A:
1.3.1 Info and Relationships

This table does not contain any `<th>` elements. If this table is used for layout, replace it with a proper layout controlled with CSS. If that's not possible, add `role="presentation"` to the table.

 Recommended Fix

```
<strong>Time</strong>
```

Test ID 147

Warning / priority 100%
Implicit table header

(line: 21, 13)

Web Content Accessibility Guidelines (WCAG) 2.0, Level A:
1.3.1 Info and Relationships

This table cell's text is visually styled to look like table headers, but does not use proper table header markup with TH elements. If these cells are intended to be headers, use TH elements.

 Recommended Fix

- Beispiel mit Formular:
 - **Tenon, aXe, Wave und HTML_CodeSniffer** melden alle Felder ohne Label

8. Praxis: Demo-Time

Demo wichtiger Webseiten und Apps mit dem Screenreader

Einschub Screenreaders: Alternativen zu VoiceOver

- NVDA (Open-Source-Alternative für Windows)
 - Download: <https://www.nvaccess.org>
 - Start: Programm starten
- Einführung: <https://www.marcozehe.de/how-to-use-nvda-and-firefox-to-test-your-web-pages-for-accessibility/>
- Tastatur-Shortcuts: <https://webaim.org/resources/shortcuts/nvda#reading>

Screenreaders: Weitere Alternativen

- Windows:
 - **Jaws** (kommerziell): <https://www.freedomscientific.com/Products/software/JAWS/>
 - **Narrator**: Einfacher Screenreader, Teil von Windows: <https://support.microsoft.com/en-us/help/17173/windows-10-hear-text-read-aloud>
- Android:
 - **TalkBack**: <https://play.google.com/store/apps/details?id=com.google.android.marvin.talkback&hl=en>
- Alle Plattformen:
 - **ChromeVox** (Extension): <https://chrome.google.com/webstore/detail/chromevox-classic-extensi/kgejglhpjiefppelpmljglcjbhoinfn?hl=en>
- Linux:
 - **Orca** (Open Source, Teil von GNOME): <https://wiki.gnome.org/Projects/Orca>

Weitere Infos: <https://webaim.org/techniques/screenreader/>



DEMO TIME

We can make a difference!

Q & A

Werner Hänggi



Werner Hänggi
werner.haenggi@adnovum.ch

[linkedin.com/in/werner-h%C3%A4nggi-716718145/](https://www.linkedin.com/in/werner-h%C3%A4nggi-716718145/)

Thomas Jaggi



Thomas Jaggi
thomas@responsive.ch

[linkedin.com/in/thomasjaggi](https://www.linkedin.com/in/thomasjaggi)

Twitter/Github: @backflip

Anleitung VoiceOver auf OS X

- Aktivierung: System Preferences > Accessibility > VoiceOver
- Ein-/Ausschalten: **Command-F5**

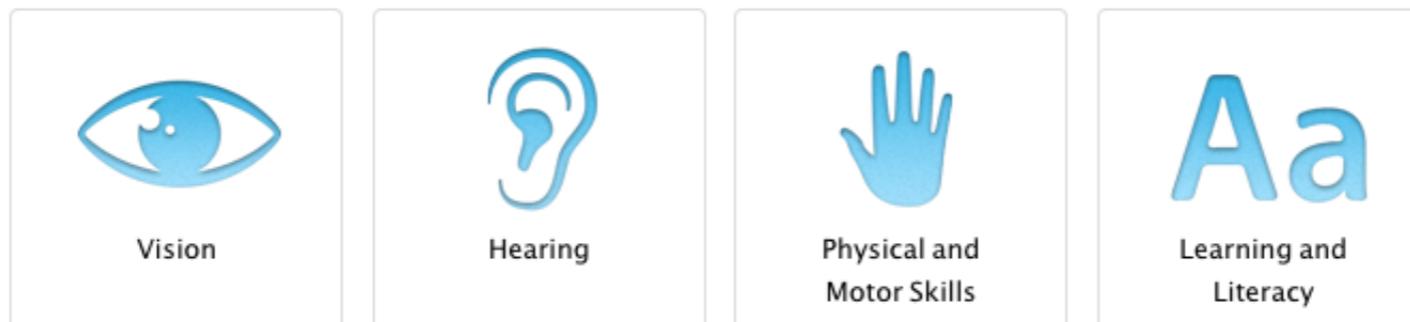
Weitere Infos: <https://www.apple.com/accessibility/mac/vision/>

Anleitung: VoiceOver auf iOS

- Aktivierung: Settings > (*General* >) Accessibility > VoiceOver
- Ein-/Ausschalten: Settings > (*General* >) Accessibility > Triple-click home > VoiceOver

iOS. A wide range of features
for a wide range of needs.

Intuitive by design, iPhone, iPad, and iPod touch also come with assistive features that allow people with disabilities to experience the fun and function of iOS. With these innovative technologies built right in, iOS devices become powerful and affordable assistive devices.



Ressourcen

- **Richtlinien:**
 - WCAG 2.1 (@w3c_wai): <https://www.w3.org/TR/WCAG21/>
 - P028: https://www.isb.admin.ch/isb/de/home/ikt-vorgaben/prozesse-methoden/p028-richtlinien_bund_gestaltung_barrierefreie_internetangebote.html
- **Consulting und Zertifizierung:**
 - Access for all (@Access4All): <https://www.access-for-all.ch>
- **Anleitungen and News:**
 - Werner: "In the Code: Was braucht's für barrierefreie Software?" <https://www.inside-it.ch/articles/56026>
 - The Paciello Group by Steve Faulkner (@stevefaulkner) & Co.: <https://www.paciellogroup.com>
 - WebAIM (@WebAIM) by Jared Smith (@jared_w_smith) & Co. : <https://webaim.org>
 - Karl Groves (@karlgroves): <https://www.karlgroves.com>
 - Marco's accessibility blog (@MarcoZehe): <https://www.marcozehe.de>
 - The Accessibility Project (@ A11YProject): <https://a11yproject.com>

Ressourcen: ARIA

- **Richtlinien:**
 - WAI-ARIA 1.1: <https://www.w3.org/TR/wai-aria/>
 - WAI-ARIA 1.1 Authoring Practices: <https://www.w3.org/TR/wai-aria-practices/>
- **Globale Helper:**
 - «ally.js» by Rodney Rehm (@rodneyrehm): <https://allyjs.io>
- **Beispiel Tabs:**
 - «Accessible ARIA Tabs» by Jason Kiss (@jskiss): <http://www.accessibleculture.org/research/aria-tabs/>
- **Beispiel Dialoge:**
 - «A11y Dialog» by Edenspiekermann: <https://github.com/edenspiekermann/a11y-dialog>
- **Beispiel Menus:**
 - «Accessible Mega Menu» by Adobe (@ AdobeAccess): <https://adobe-accessibility.github.io/Accessible-Mega-Menu/>